AISR NNG06GE59G

On-the-fly and Grid Analysis of Astronomical Images for the Virtual Observatory

PI:
Andrew Ptak
Johns Hopkins University

Co-I:
Andrew Connolly
University of Washington

Introduction

This AISR project combines two previous AISR-funded efforts, XAssist and WESIX. XAssist (http://www.xassist.org) is a program that automatically analyzes *Chandra*, *XMM-Newton*, and *ROSAT* data. XAssist is particularly adept at processing large numbers of datasets, and accordingly has been used to produce pipelines for these missions. The *Chandra* and *XMM-Newton* pipeline source lists are included as tables that can searched at the HEASARC (http://heasarc.gsfc.nasa.gov). WESIX (Web Enabled Source Identification with XMatching; http://nvogre.astro.washington.edu:8080/wesix/) is a web service that runs the source-detection program SExtractor on an uploaded image. The resultant source list is then cross-matched with selected catalogs. The main goals of this project are to develop a set of web services for the XAssist functionality to allow X-ray analysis to be done on-the-fly and in a distributed fashion, and to expand the WESIX web service to also include X-ray images. This is a natural pairing since WESIX is an existing web service that is already part of the Virtual Observatory for non-interactive analysis of optical images, and XAssist has been developed from the beginning for non-interactive usage.

Status

XAssist

The original XAssist web site was driven by Java (specifically Java Server Pages) however maintenance (e.g., adding news items) was tedious. A new web site is in place that is running on the content management system (CMS) Plone. This has allowed us to post news and documentation much more easily than before. The installation and “theme” creation for this was done by a graduate student employed by the project. We originally intended to have Plone also serve as a web application framework, however it is evidently too complex for this to be the case easily. We are instead in the process of writing web applications, specifically to allow for user requests of fields and to query the properties of a specific source or region, in Django. Django is a web application framework that is well-suited for this task. We are using style-sheets from the Plone site to mimic the look-andfeel of the main Plone web site for the web applications (ideally users will not be aware of the two different platforms), and potentially in the future we will convert the entire site (i.e., both content management and web applications) to be under a single Django system. This was not done from the start since Django is a low-level application framework, and content management functionality would have required a significant programming effort or investigation into available Django modules that could be "plugged in". Now that we have gained some experience with Django, either of these options are more feasible now. Therefore we will try existing CMS modules at first and then create our own as necessary (if at all).

A major improvement in the XAssist pipeline code is that it is now controlled by a proper queuing system. Before this was completed, the pipeline processing was driven by simple shell scripts. This did not allow for any prioritization and required new fields to be added manually. The scripts often had to be restarted to be aware of new fields or to re-attempt processing of problematic fields. The queue system consists of a main controller program that manages requests and client programs that request fields for processing and then run xassist on those fields. A Django interface to controlling the queue has been implemented, currently intended for administration, and we have developed initial Django code to allow general users to submit requests as mentioned above. Scripts have also been written to systematically add jobs to the queue (through crontab jobs) to keep the processing running continuously. Web services that had been integrated into the xassist code itself are being used to report the status of the processing of a given field to be used when the Django queue web application reports on the processing status.

We are continuing to refactor the XAssist code base to improve its efficiency and to allow individual functions to be isolated. The current version has a command-driven interface that will in turn allow for the development of a graphical user interface. Since there will in general be a correspondence between user options and controls in a graphical interface and web services, we have been developing an external control program that "attaches" to running instances of XAssist. An initial version of this program is available now on the XAssist web site. A key feature is being able to search the pipelines that are running on the XAssist web site and to download products (e.g., images of the fields). The ability to submit processing requests has recently been added and is being tested. Later versions that will allow precise control over processes running on a local network will be particularly useful in a grid environment. This will allow users to monitor the process of XAssist jobs running on different computers.

We have added the capability to use SExtractor on X-ray data. We are calibrating the use of SExtractor for other missions and early results are promising.

WESIX

The co-I A. Connolly is supervising work on WESIX. We have begun to develop an updated interface to WESIX to allow for the submission of X-ray fields as well as optical images. The main modification will be to add parameters that are relevant to X-ray data, which of course will be ignored when only an optical image is uploaded. The WESIX project is has transitioned from using SOAP to XML-RPC. The WESIX team, now at the

University of Washington, has transitioned to using Python as the underlying language. This will further enhance the synergy between XAssist and WESIX. WESIX has also been updated to allow multi-band photometry (i.e., sources can be detected in one image with photometry derived from a different image, typically using other filters).

Current Work

The main drive of current work is finalizing a coherent web service interface, now using XML-RPC, that is equally applicable to both XAssist and WESIX. The current set up will include defaults for either optical or X-ray data, allowing the user to specify minimal parameters in typical cases. Advanced users can specify parameters via a dictionary (i.e., associative array) parameter type. The adoption of both XML-RPC and Python by both projects is simplifying the development. We are keeping in mind, of course, that the spirit of web services is interoperability across programming languages, and therefore we will test clients written in Java, and ask for volunteers to try clients in other languages (such as Perl, C/C++ or C#). We have been working on protoype functions to allow WESIX to query XAssist and allow it dispatch job requests.

There are four main bottlenecks in the processing of X-ray data by XAssist: 1) reprocessing the data to incorporate the latest calibration, 2) source detection, 3) fitting the image of each source in order to ascertain the spatial extent and to determine efficient source extraction regions for spectra, and 4) generating instrumental responses for each source to properly determine the flux, and, for sources with enough counts, to perform spectral fitting. These steps help to make results of XAssist be more reliable, but take too much time for an on-the-fly processing system. We are continuing with our original plan to allow for quicker response times.

If a requested field has already been processed, then the pipeline data can be accessed directly and these bottlenecks are obviously avoided. If not, but the data for the field requested is public or uploaded by the user, then two jobs will be spawned. The first will perform quick-look analysis of the data where reprocessing is not done (and is often only a marginal improvement). SExtractor can now be used for source detection, which runs in seconds. The spatial fitting step will also be skipped and it will be assumed that sources are unresolved, i.e., model point-spread functions will be used for the source extent. Finally, pre-computed responses will be used to derive fluxes. The second job will add the field to the XAssist pipeline queue with high priority to perform full processing. XAssist also has been modified to allow only a subset of a field to be processed if requested, which of course will shorten the processing time where the user is only interested in a specific source or region.

Future Work

We anticipate the development of web services for the basic functionality of XAssist by the summer of 2010, namely web services to request the full processing of a dataset, to retrieve the properties of sources within a given region of a processed field (if any have been detected), to compute upper-limits at a given position, and to perform quick-look analysis of a given dataset (to be directly incorporated into the WESIX web service). Over this time period we will continue to improve the performance of both XAssist and WESIX separately. Both the quick-look and full processing functionality will be implemented for *Suzaku* data, and, time permitting, also *Swift* XRT data.

The current status of XAssist was presented in at the High-Energy Astrophysics Division (HEAD) meeting in March, 2010. We also plan on submitting a paper to ApJS detailing the combined XAssist-WESIX functionality by the end of 2010.